



ATTORNEY DOCKET NO.
1308.0419

PATENT APPLICATION
09/972,365

1

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Ulrich Bungert et al.
Serial No.: 09/972,365
Filing Date: October 5, 2001
Examiner: Michael Maskulinski
Art Unit: 2113
Title: Automatic Generation of Diagnostic
Programs for SPS-Controlled Systems

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

BEST AVAILABLE COPY

Dear Sir:

DECLARATION PURSUANT TO 37 C.F.R. § 1.131

I, Ulrich Bungert hereby declare and state that:

1. I am the co-inventor of the subject matter of the above-referenced Application entitled "Automatic Generation Of Diagnostic Programs For SPS-Controlled Systems" filed on November 9, 20021 (the "Application") claiming priority to provisional application 60/318,180 filed September 7, 2001.

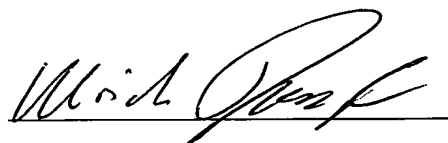
2. The Examiner rejected Claims 1-13 of the Application in an Office Action dated February 8, 2006 based, in part, on Lenz et al. (Pub. No. US2001/0032025) filed February 14, 2001 (the "Lenz Application") and claiming priority to provisional application 60/182,247 filed February 14, 2000 ("the critical date") or in the alternative based, in part, on Ramadei et al (Pub. No. US2002/0166082) filed March 2, 2001 (the "Ramadei Application").

3. I developed an understanding and appreciation of the subject matter of Claims 1-13 of the Application prior to the critical date, the priority date of the Lenz Application. Attached herewith as Exhibit A is a German Invention Disclosure Form prepared by me, which was received on June 17, 1998, detailing my understanding of the subject matter of the claims. A correct translation of this Invention Disclosure Form into English is also submitted.

4. From a point prior to critical date continuing through to the filing date of the Application, I worked with a patent attorney retained by my employer to prepare papers for filing. That work included conducting a patentability search and reviewing the resulting references, discussing the invention and possible claims, preparing a draft application, reviewing the draft application and preparing suggested changes, reviewing a revised application, preparing a declaration, reviewing and signing the declaration, and preparing the filed application.

5. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true. Further, I declare that these statements are made with the knowledge that willful false statements, and the like so made, are punishable by fine or imprisonment, or both, under Section 1001, Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the Application or any patent issuing thereon.

Signed this 2nd day of Mai, 2006.



Ulrich Bungert

Model-based Diagnosis in Manufacturing Technology

1. Introduction

For the operator of the machines and systems that are becoming more and more complex, achieving an efficient operation (techn. availability, output, . . .) is of decisive importance. Strategies for this are required which permit quick and effective intervention in case of malfunction.

Here special importance must be attached to system diagnosis.

In the scope of a prototype study a demonstrator for model-based diagnosis (MBD) was developed. For this, diagnostic methods that were developed at ZT SE4 were drawn upon. By means of these methods a real machine component was modeled and objects of diagnostic realized based on it.

It was the goal of the investigation to demonstrate the technical feasibility of the approach and to find out what potential platforms can be drawn on for the use of this technology. A significant component was to highlight the fact that the user does not have to program the model-based diagnosis but rather interconnects the necessary modules with the process via the signals to be measured.

2. Demonstration example

The example chosen for this object is a section of a production line. It is the catch cup component of a screw station at AT ER. This component has the object of traversing a mechanism by means of pneumatic drives from a base position into an operating position and back.

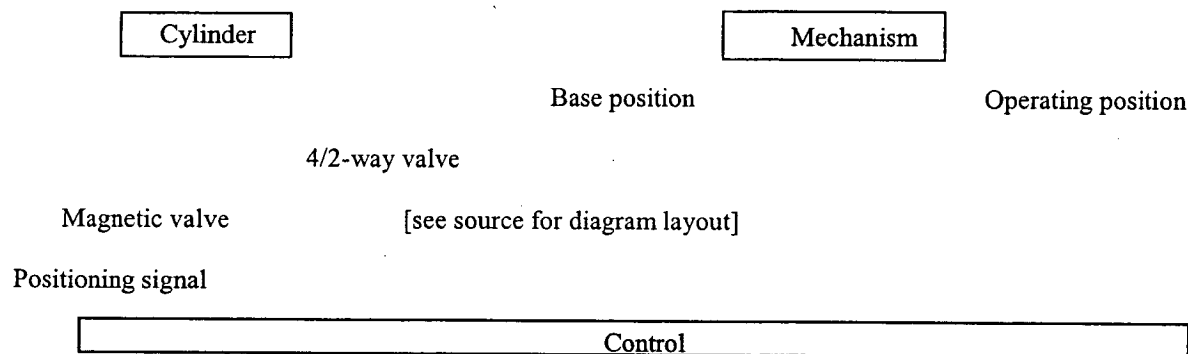


Figure 1: Pneumatic drive

The catch cup is a safety device in automatic screwing. It consists of two pneumatic drives according to figure 1. Such a pneumatic drive is actuated via the magnetic valve. The directional valve engages and lets the compressed air flow into the left chamber of the cylinder, which thereupon traverses into position.

3. Control modules

Components of this type are customarily coordinated by process controls. In the example, simple control modules were programmed. These were interconnected to a process environment so that the diagnostic modules can be linked and tested.

4. Model-based diagnosis

Model-based diagnosis provides technologies that perform a continuous theoretical-actual comparison of the machine process and here calculate the probabilities of faults of the components integrated in the model. For this, explicit process models were used that contain a description of the behavior of the process.

The process modules consist of modules that are in turn typed. An actual system is described by instances of these types. Thus it is possible, for example, to introduce the sensor component since the sensor for registering the base position and that for registering the operating position behave in the same manner.

In the following, the example (from figure 1) is represented as a block diagram. There the components have already been marked with the possible faults.

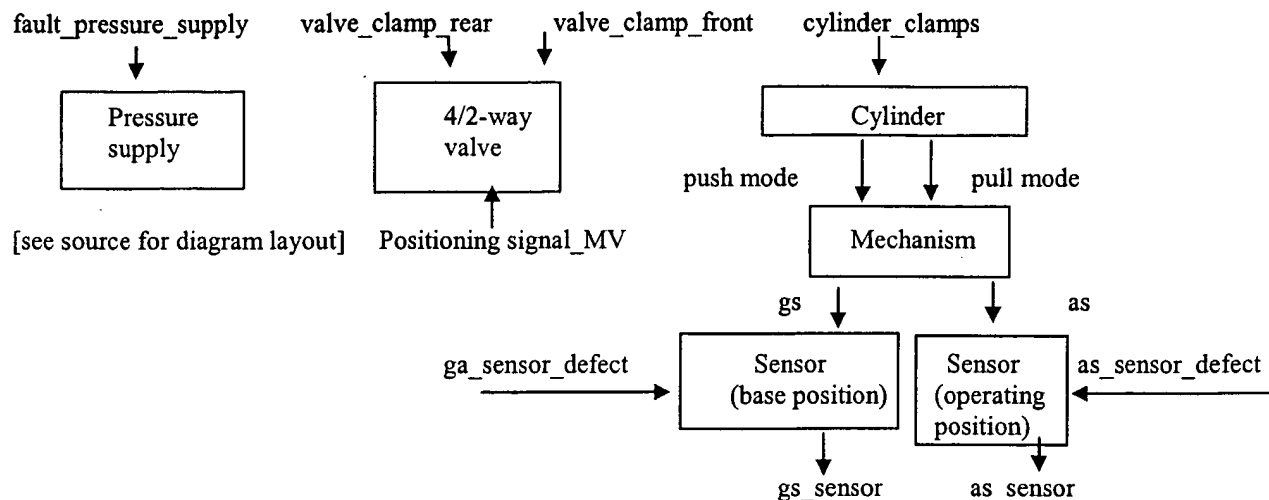
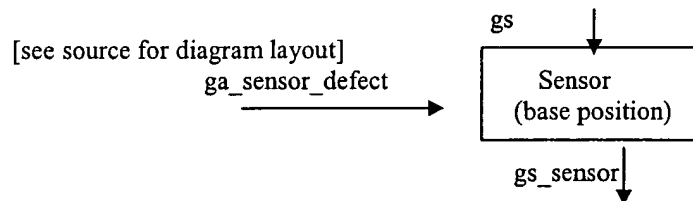


Figure 2: Block diagram of the example

For each component a fault model must be stored that describes the connection between the input and output quantities:



The sensor sends in faultless operation ($\neg gs_sensor_defect$) a 1 (gs_sensor) if the mechanism is in the base position (gs):

$gs \ \& \ \neg gs_sensor_defect \implies gs_sensor$.

If the mechanism is outside of the base position ($\neg gs$) or if a sensor defect occurs (gs_sensor_defect), a 0 is sent ($\neg gs_sensor$).

$\neg gs \implies \neg gs_sensor$.

$gs_sensor_defect \implies \neg gs_sensor$.

The output of the sensor is available as a minimum value.

measurablesignals [gs_sensor].

With the achieved granularity of the model there is a sufficiently precise process model to be able to make diagnostic statements up to the component level (cylinder, sensor, . . .). The modeling technology used offers the capability of realizing different degrees of freedom. The models have an interface via which the actor and sensor signals are registered.

5. Diagnostic modules

From the configured process model an executable diagnosis program (implemented in C++ or SCL) is generated automatically.

The diagnosis program takes over the diagnosis-side progression through the theoretical process steps and the calculation of the results of the diagnosis. The complete model of the pneumatic drive was developed in the course of the study and from it a C++ source and a SCL source were generated. The C++ source was prepared as an executable DLL. The SCL source was prepared in STEP 7 and executed on a SIMATIC S7-300 and in S7-PLCSIM.

SIEMENS

Notice of Invention

Title

**Automatic Generation of Diagnosis Programs for SPS-controlled
Systems**

Automatic Generation of Diagnosis Programs for SPS-controlled Systems

Problem:

In the framework of process control systems great importance must be attached to diagnosis. On the one hand, a malfunction must be recognized within the shortest time so that damaging aftereffects can be prevented. On the other hand, the cause of the fault must be identified precisely so that the repair times, and consequently the downtimes, are as short as possible.

In order to be able to realize the objects of the diagnosis, knowledge of the process is needed. Previously, explicit diagnostic rules have frequently been formulated and applied in diagnostic systems, i.e. IF-THEN rules of the form "IF measured values THEN fault." In so doing, a major problem is systematic rule generation, especially in complex industrial systems.

The knowledge of the process is frequently only incomplete and uncertain. It is necessary to use a correct formalism to register and handle this uncertainty.

As a rule, the results of diagnosis are ambiguous because different faults can give rise to the same measurements. On account of this, a weighting of different faults is desirable.

The effects of the faults are not always measurable at the time they occur but rather only make themselves noticeable at a later time. Such delays must be taken into account in the knowledge of the process as well as in the realization of the object of the diagnosis.

An additional important criterion for the use of diagnosis systems is that they can be incorporated economically into existing hardware and software of process control technology.

Solution:

The essential features of the diagnosis developed are:

1. Use of qualitative models:

The models for the diagnosis can be formulated very roughly. Instead of the precise description of the relationships between the signals, logical model formulas, such as, for example, IF-THEN rules, are often adequate:

IF causes THEN effects,

which are expressed here in the following notation:

Values_of_the_input_signals \implies Values_of_the_output_signals

These models necessary for diagnosis represent a very rough description of the normal or faulty process behavior and can be developed with relatively little effort. However, the direction of action described in the models (values of the input signals and the faults cause values of the output signals) do not correspond to the final direction of inference for the realization of the objects of diagnosis (values of the measurable signals lead to statements about faults).

2. Extension to dynamic relationships:

Using only static relationships is rarely sufficient since certain signal configurations only have as consequences effects at the next point in time. As the model forms, a clocked automaton was chosen which describes how the instantaneous values of the input signals and the instantaneous state cause the values of the output signals and the following instantaneous state. The current point in time is marked by k , the one following that by $k + 1$:

Values_of_the_input_signals(k) & state(k) \implies
Values_of_the_output_signals(k) & state($k+1$).

3. Extension to fault models:

An extension that occasions a slightly greater complexity in modeling is the use of fault models. In these model forms the faults are incorporated on the condition side, that is, the faulty behavior for different faults is incorporated into the model:

Values_of_the_input_signals(k) & state(k) & fault(k) \implies
Values_of_the_output_signals(k) & state($k+1$).

4. Extension to uncertain models:

The causal relationships in the logical model can be enhanced by causally related probabilities.

Values_of_the_input_signals(k) & state(k) & fault(k) \implies
Values_of_the_output_signals(k) & state($k+1$)

/ $P(\text{Values_of_the_output_signals}(k) \ \& \ \text{state}(k+1) \mid$
Values_of_the_input_signals(k) & state(k) & fault(k)).

In this way it can be expressed that certain outputs only occur with a certain frequency.

5. Use of component libraries and hierarchies:

As a rule, a system model is constructed with an orientation toward components. In so doing, a component can represent

- A control/regulation function that is intended to run on an automation device or
- a physical, technological process component.

These components are typed. An actual system is described by instances of these types (type-instance concept).

Furthermore, components are structured hierarchically, that is, they can themselves be constructed from other components.

Each component is described by an automaton expressed in notation in the form of logical formulas.

The modeling for the diagnosis can frequently be accomplished with relative simple means. It can, for example, be done in parallel to the design of the control unit.

6. Division of the diagnosis algorithm into off-line and on-line phases:

The object of the diagnosis consists of determining, from the measured values, the faults occurring in the process:

Diagnosis algorithm:

Given: Measured value sequences,
Model,

Sought: Fault

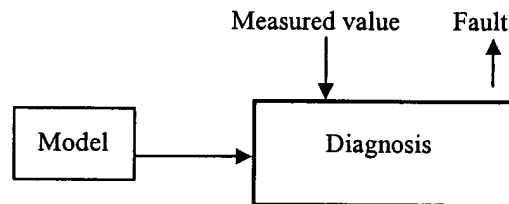


Figure 1 Fundamental schema of the diagnosis

This objective can be realized with various methods. However, the realization of this object can frequently be time-critical so that the computational steps taking place at the time of diagnosis must be optimized. On account of this there is a division of the diagnosis algorithm into an off-line and on-line phase. In the off-line phase a diagnosis program is generated from the process model, where the diagnosis program enables the inference, from the measured values, of faults under real-time conditions in the on-line phase.

In the on-line phase the diagnosis program is generated from the system model and corresponding specifications for the hardware and software environment of the on-line phase.

Diagnosis algorithm:

Given: Model,
Specifications for the hardware and software environment

Sought: Diagnosis program

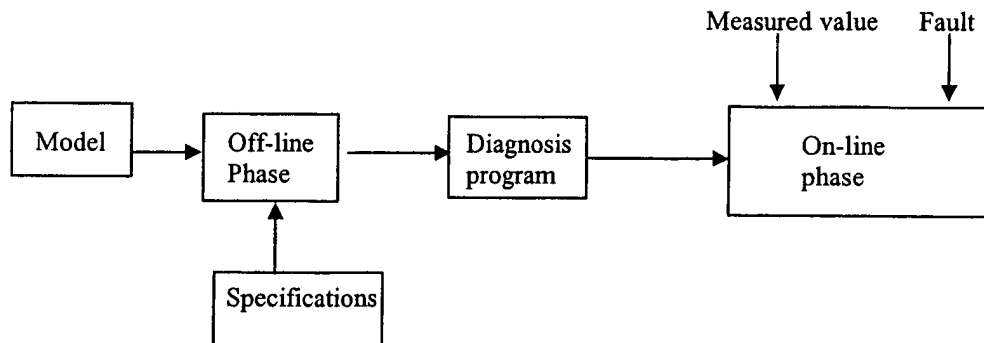


Figure 2 Subdivision of the diagnosis into on-line and off-line phases

This diagnosis program then executes the diagnosis in the system environment (on-line phase):

On-line phase:

Given: Measured values,
Diagnosis program,

Sought: Faults

The off-line phase is realized with the logical programming language Prolog. The diagnosis program is formed according to the specifications in the syntax of an imperative programming language. At present programs can be generated in SCL and C++.

7. Calculation of the probability of faults occurring:

Instead of the output of four alternative solutions that consist of combinations of faults, there is a calculation of the probabilities of occurrence.

The solution consists of the associated probability for each fault that it occurred under the condition of the measurements that occurred previously (0 ... k):

$$P(\text{fault}(k) \mid \text{Values_of_the_output_signals}(0 \dots k) \& \text{Values_of_the_input_signals}(0 \dots k)).$$

As an intermediate result the calculation of

$$P(\text{fault}(k) \& \text{state}(k+1) \mid \text{Values_of_the_output_signals}(0 \dots k) \& \text{Values_of_the_input_signals}(0 \dots k)).$$

is necessary for this and done recursively.

8. Calculation of the probability of faults occurring in the past:

Corresponding to process dynamics, faults often do not have an effect at the moment of their occurrence but rather only after a certain time and then can only be recognized based on the measurements. On account of this, it is calculated for each fault with what probability it occurred sometime in the past:

$$P(\text{fault}(k) \vee \text{fault}(k-1) \vee \dots \vee \text{fault}(0) \mid \\ \text{Values_of_the_output_signals}(0 \dots k) \& \text{Values_of_the_input_signals}(0 \dots k)).$$

As an intermediate result the calculation of

$$P(-\text{fault}(k) \& \dots \& -\text{fault}(0) \& \text{state}(k+1) \mid \\ \text{Values_of_the_output_signals}(0 \dots k) \& \text{Values_of_the_input_signals}(0 \dots k) \& \\ -\text{fault}(k) \& \dots \& -\text{fault}(0)).$$

is necessary for this and done recursively.

9. Calculation of the probability of a composite fault:

As a summary of the probabilities of occurrence of faults the probability of a composite fault is calculated, that is, the probability that any fault occurs. Only when this probability assumes the value 1 must the probabilities of the individual faults be considered in more detail.

Seiten / Anlagen
4/-

Dateiname
KurzDoc_MBD.doc

Dokumenten-Nr.:
A&D GT4/98-02

Projekt: Modellbasierte Diagnose in der Fertigungstechnik		Projektart: Vorfeldstudie
Unterlage: Kurzbeschreibung		
Version: V1.0	Datum: 26.1.98	Freigabe:
	Datum	Unterschrift

Bearbeiter:	Dienststelle:
Hr. Bungert	A&D GT4
Dr. Schiller	ZT SE4

Verteiler:	Dienststelle:
Hr. Beck	A&D AS E1
Hr. Friedrich	A&D GT4
Hr. Hammon	A&D AS S1
Dr. Gaißmaier	A&D AS E1
Hr. Lange	A&D AS S8
Dr. Lo	A&D AS S8
Hr. Schmoll	A&D AS S8
Fr. Schulz	A&D AS S1

Inhaltsverzeichnis

Inhaltsverzeichnis.....	1
1. Einleitung	2
2. Demonstrationsbeispiel.....	2
3. Steuerungsbausteine.....	2
4. Modellbasierte Diagnose	2
5. Diagnosebausteine	3
6. Erprobungsumgebung.....	4
7. Ergebnisse und Ausblick	4

1. Einleitung

Für die Betreiber der immer komplexer werdenden Maschinen und Anlagen ist die Erzielung eines effizienten Betriebs (techn. Verfügbarkeit, Ausbringung, ...) von entscheidender Bedeutung. Es sind hierzu Strategien erforderlich, die es gestatten, im Störfall schnell und wirkungsvoll einzugreifen. Der Anlagen-Diagnose kommt hier eine besondere Bedeutung zu.

Im Rahmen einer Vorfeld-Studie wurde ein Demonstrator für die Modellbasierte Diagnose (MBD) entwickelt. Es wurden hierfür Diagnosemethoden, die bei ZT SE4 entwickelt wurden, herangezogen. Mittels dieser Methoden wurde eine reale Maschinenkomponente modelliert und darauf aufbauend Diagnoseaufgaben gelöst.

Ziel der Untersuchung war es, die technische Machbarkeit der Vorgehensweise nachzuweisen und herauszufinden, welche potentiellen Plattformen für den Einsatz dieser Technik herangezogen werden können. Ein wesentlicher Bestandteil war aufzuzeigen, daß der Anwender die Modellbasierte Diagnose nicht programmieren muß, sondern die benötigten Bausteine über die zu messenden Signale mit dem Prozeß verschaltet.

2. Demonstrationsbeispiel

Das für diese Aufgabe gewählte Beispiel ist ein Ausschnitt aus einer Fertigungslinie. Es handelt sich dabei um die Komponente Auffangbehälter einer Schraubstation bei AT ER. Diese Komponente hat die Aufgabe, eine Mechanik mittels pneumatischer Antriebe von einer Grundstellung in eine Arbeitsstellung und zurück zu verfahren.

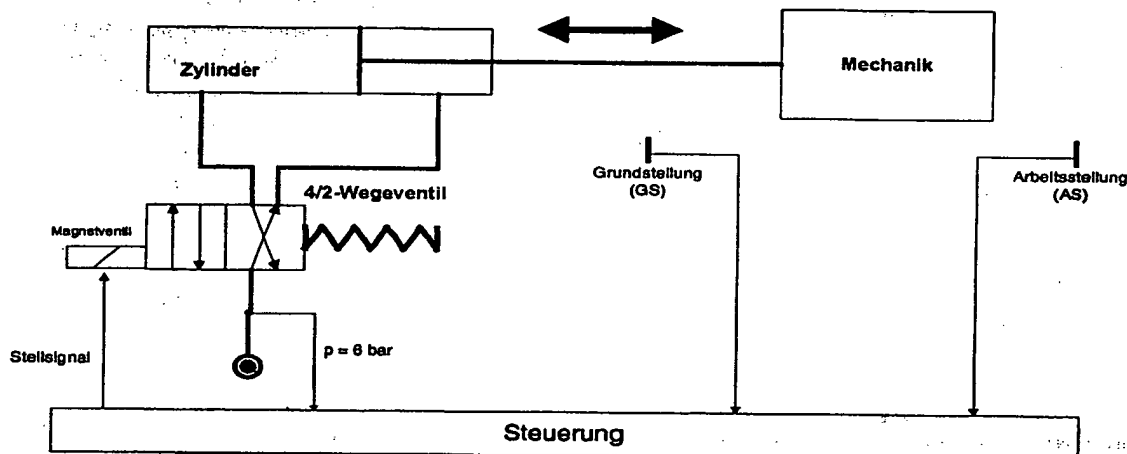


Bild 1: Pneumatischer Antrieb

Der Auffangbehälter ist eine Sicherheitsvorrichtung beim automatischen Verschrauben. Er besteht aus zwei pneumatischen Antrieben gemäß Bild 1. Ein solcher pneumatischer Antrieb wird über das Magnetventil angestoßen. Das Wegeventil schaltet und läßt die Druckluft in die linke Kammer des Zylinders strömen, der daraufhin ausfährt.

3. Steuerungsbausteine

Derartige Komponenten werden üblicherweise durch Ablaufsteuerungen koordiniert. Für das Beispiel wurden einfache Steuerungsbausteine programmiert. Diese wurden zu einer Ablaufumgebung verschaltet, so daß die Diagnosebausteine angebunden und erprobt werden können.

4. Modellbasierte Diagnose

Die Modellbasierte Diagnose stellt Techniken bereit, die einen stetigen Soll-Ist-Vergleich des Maschinenablaufs durchführen und hier die Fehlerwahrscheinlichkeiten der im Modell integrierten Komponenten berechnen. Dafür wurden explizite Prozeßmodelle verwendet, die eine Beschreibung des Prozeßverhaltens beinhalten.

Die Prozeßmodelle bestehen aus Komponenten, die wiederum typisiert sind. Eine konkrete Anlage wird durch Instanzen dieser Typen beschrieben. So ist es beispielsweise möglich, die Komponente Sensor einzuführen, da sich sowohl der Sensor zum Erfassen der Grundstellung als auch der zum Erfassen der Arbeitsstellung gleich verhalten.

Nachfolgend wird das Beispiel (aus Bild 1) als Blockschaltbild dargestellt. Dort sind bereits die Komponenten mit den möglichen Fehlern markiert.

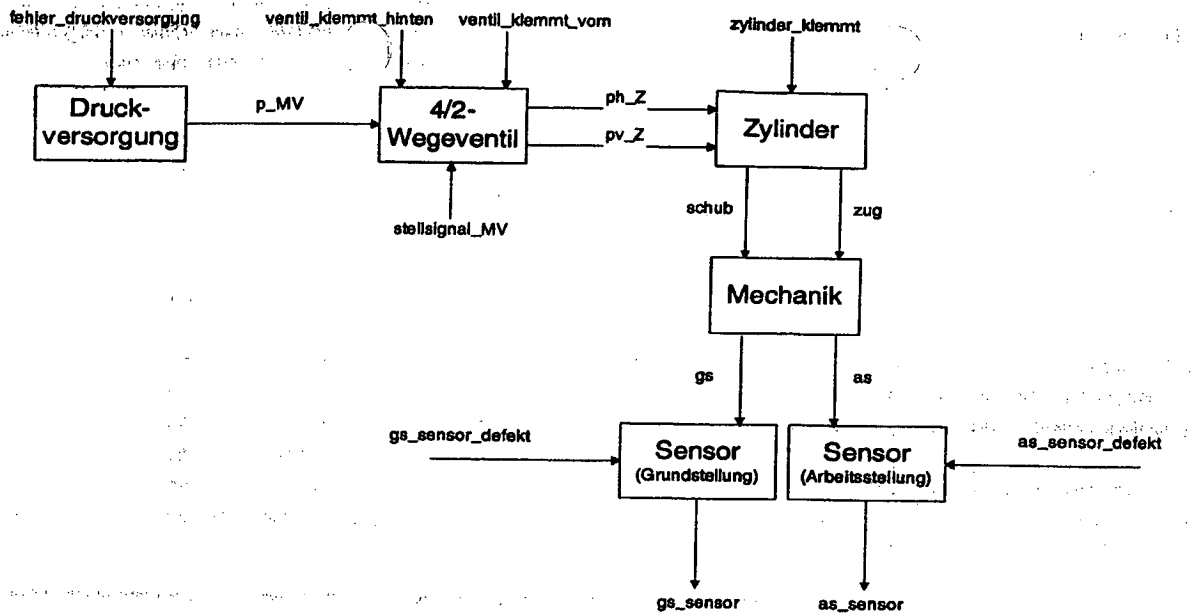
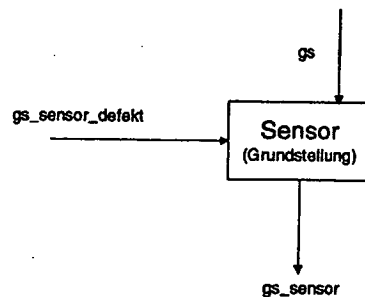


Bild 2: Blockschaltbild des Beispiels

Für jede Komponente muß ein Fehlermodell hinterlegt sein, das den Zusammenhang zwischen Eingangs- und Ausgangsgrößen beschreibt:



Der Sensor liefert im ungestörten Betrieb ($\neg gs_Sensor_defekt$) eine 1 (gs_Sensor), wenn die Mechanik sich in der Grundstellung befindet (gs):

$gs \ \& \ \neg gs_Sensor_defekt \implies gs_Sensor$.

Befindet sich die Mechanik außerhalb der Grundstellung ($\neg gs$) oder tritt ein Sensordefekt (gs_Sensor_defekt) auf, wird immer eine 0 geliefert ($\neg gs_Sensor$):

$\neg gs \implies \neg gs_Sensor$.

$gs_Sensor_defekt \implies \neg gs_Sensor$.

Die Ausgabe des Sensor steht als Meßwert zur Verfügung.

`measurableSignals [gs_Sensor].`

Mit der erreichten Granularität der Modellierung liegt ein hinreichend genaues Prozeßmodell vor, um Diagnoseaussagen bis in die Komponentenebene (Zylinder, Sensor, ...) treffen zu können. Die eingesetzte Modellierungstechnik bietet die Möglichkeit, unterschiedliche Feinheitsgrade zu realisieren. Die Modelle besitzen eine Schnittstelle, über die Aktor- und Sensorsignale erfaßt werden.

5. Diagnosebausteine

Aus dem konfigurierten Prozeßmodell wird ein ablauffähiges Diagnoseprogramm (implementiert in C++ oder SCL) automatisch generiert.

Das Diagnoseprogramm übernimmt die diagnoseseitige Weiterschaltung der Soll-Prozeßschritte und die Berechnung der Diagnoseergebnisse. Es wurde im Rahmen der Studie das vollständige Modell des pneumatischen Antriebs erstellt und daraus sowohl eine C++-Quelle als auch eine SCL-Quelle generiert. Die C++-Quelle wurde als ablauffähige DLL bereitgestellt. Die SCL-Quelle wurde in STEP 7 bereitgestellt und sowohl auf einer SIMATIC S7-300 als auch in S7-PLCSIM zum Ablauf gebracht.

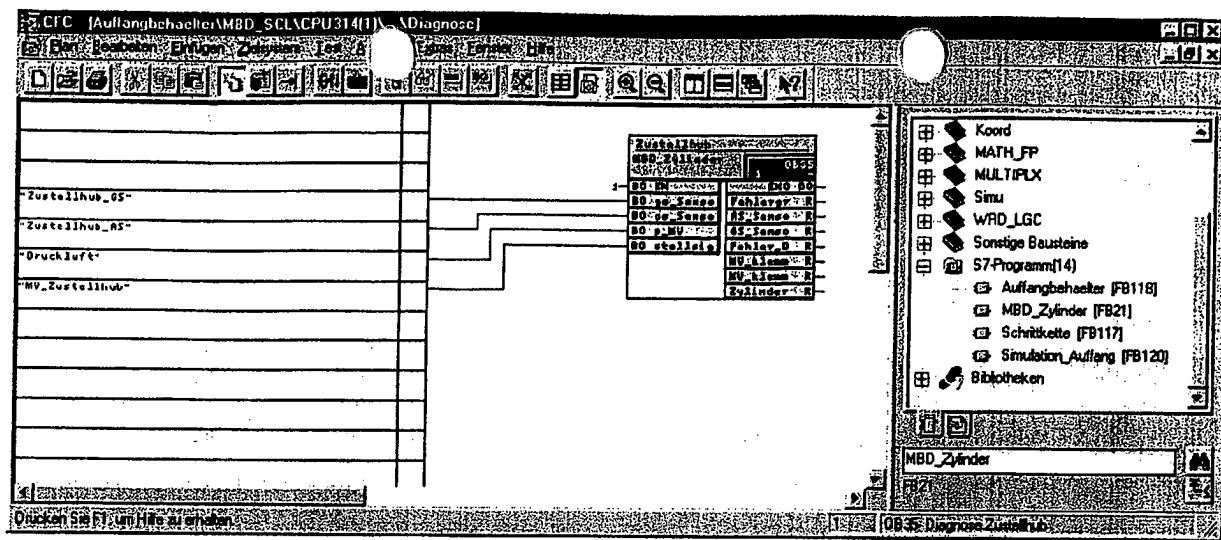


Bild 3: Diagnosebaustein im CFC

6. Erprobungsumgebung

Im letzten Schritt wurde eine einfache Ablaufumgebung, die den Maschinenablauf simuliert, aufgebaut. Die Diagnosebausteine wurden mit dem damit vorliegenden virtuellen Prozeß verschaltet.

7. Ergebnisse und Ausblick

Die durchgeführten Erprobungen stellen die Basis für die weiterreichende Untersuchung der Einsatztauglichkeit dieser Technik dar. Es wurde zunächst festgestellt, daß die Diagnosebausteine auf SIMATIC-S7 und S7-PLCSIM dieselben Ergebnisse berechnen, wie die auf PC ablaufenden Diagnosemodule in Form von C++-DLLs.

Der Komponentenansatz zeigt die Machbarkeit einer Diagnose-Standardbaustein-Bibliothek, die mit kleinem Typvorrat einen großen Anteil an Automatisierungslösungen abdecken kann. Diese Bibliothek kann hierarchisch gegliedert sein und sowohl Basistypen (Ventil, Motor, ...) als auch aggregierte, komponentenorientierte Typen (pneumatischer Antrieb, ...) beinhalten. Die Diagnosemodule können potentiell als technologische Funktionen oder im Rahmen von Diagnose-Faceplates zum Einsatz kommen.

Die Einsetzbarkeit der Diagnosebausteine auf einer SIMATIC S7 ist begrenzt aufgrund der zulässigen Maximalgröße von Bausteinen beim Runterladen. Bei größeren Bausteinen kann zwar eine Aufteilung vorgenommen werden, die wäre allerdings wenig effizient und würde zudem den komponentenorientierten Ansatz verletzen. Weiterhin darf die Belastung des Zyklusses bei zusätzlichen Bausteinen nicht vernachlässigt werden.

Die Technik der Modellbasierten Diagnose stellt für die Object Engine und WinAC eine interessante Ergänzung dar. Die Tragfähigkeit dieser Vorgehensweise muß in einer realen Umgebung untersucht werden. Insbesondere der Abdeckungsgrad notwendiger technologischer Komponenten durch verschaltbare Diagnosebausteine sowie Konformität zur PCS7-Bibliothek müssen in einem nächsten Schritt betrachtet werden. In diesem Zusammenhang ist der Erstellungsaufwand einer Diagnose-Standardbaustein-Bibliothek zu ermitteln.

SIEMENS

Erfindungsmeldung

Titel

**Automatische Generierung von Diagnoseprogrammen für
SPS-gesteuerte Anlagen**

Vertraulichkeitsgrad
Nur für internen Gebrauch

Bereich
A&D
Herausgebende Dienststelle
GT 4

Aktenzeichen

Textseiten / Anlagen

Ort und Datum
Nbg M, 17.06.98

Verfasser

Dienststelle

Fernsprecher

Gegenzeichnung

U. Bungert

A&D GT 4

4814

Dr. F. Schiller

A&D GT 4

3462

Verteiler

Hr. Friedrich

A&D GT 4

Zusammenfassung

Proprietary data. company confidential. All rights reserved.
Confite a titre de secret d'entreprise. Tous droits réservés.
Comunicado como segredo empresarial. Reservados todos los derechos.
Confiado como secreto industrial. Nos reservamos todos los derechos.

Weitergabe sowie Vervielfältigung dieser Unterlage, Vervielfältigung und Mitteilung ihres Inhalts ist nicht zulässig.
Weitergabe und Mitteilung ihres Inhalts ist nicht zulässig.
nicht ausdrücklich zugestanden. Zuwiderhandlungen sind
pflichten zu Schadensersatz. Alle Rechte vorbehalten. Insbe-
sondere für den Fall der Patenterteilung oder GM-Eintragung.

Automatische Generierung von Diagnoseprogrammen für SPS-gesteuerte Anlagen

Problem:

Im Rahmen von Prozeßleitsystemen kommt der Diagnose eine besondere Bedeutung zu. Einerseits muß innerhalb kürzester Zeit ein Fehlverhalten erkannt werden, damit Folgeschäden verhindert werden können, andererseits muß die Fehlerursache genau identifiziert werden, damit die Reparatur- und demzufolge die Stillstandszeiten möglichst kurz sind.

Um Diagnoseaufgaben lösen zu können, wird Wissen über den Prozeß benötigt. Bisher werden in praktischen Diagnosesystemen häufig explizite Diagnoseregeln formuliert und angewendet, d.h. WENN-DANN-Regeln der Form „WENN Meßwerte DANN Fehler.“ Ein großes Problem ist dabei eine systematische Regelgenerierung, speziell bei komplexen industriellen Anlagen.

Das Prozeßwissen ist häufig nur unvollständig oder unsicher bekannt. Es ist notwendig, einen korrekten Formalismus zur Erfassung und Verarbeitung dieser Unsicherheit einzusetzen.

Die Diagnoseergebnisse sind in der Regel mehrdeutig, weil verschiedene Fehler gleiche Messungen verursachen können. Erstrebenswert ist deswegen eine Wichtung verschiedener Fehler.

Die Wirkungen der Fehler sind nicht immer zum Zeitpunkt ihres Auftretens meßbar, sondern machen sich erst mehrere Zeitpunkte später bemerkbar. Solche Verzögerungen müssen sowohl im Prozeßwissen als auch bei der Lösung der Diagnoseaufgabe berücksichtigt werden.

Ein weiteres wichtiges Kriterium für den Einsatz von Diagnosesystemen ist, daß sie kostengünstig in bestehende Hard- und Software der Leittechnik eingebunden werden können.

Lösung:

Die wesentlichen Merkmale der entwickelten Diagnosemethode sind:

1. Einsatz qualitativer Modelle:

Die Modelle für die Diagnose können sehr grob formuliert werden. Anstelle der genauen Beschreibung der Beziehungen zwischen den Signalen reichen oft logische Modellformeln wie zum Beispiel WENN-DANN-Regeln:

WENN Ursachen DANN Wirkungen,

die hier in folgender Form notiert werden:

Werte_der_Eingangssignale ==> Werte_der_Ausgangssignale.

Diese zur Diagnose notwendigen Modelle stellen eine sehr grobe Beschreibung des normalen bzw. fehlerhaften Prozeßverhaltens dar und sind mit relativ geringem Aufwand zu erstellen. Die in den Modellen beschriebene Wirkungsrichtung (Werte der Eingangssignale und der Fehler verursachen Werte der Ausgangssignale) entspricht jedoch nicht der letztendlichen Schlußfolgerungsrichtung zur Lösung von Diagnoseaufgaben (Werte der meßbaren Signale führen auf Aussagen über Fehler).

2. Erweiterung auf dynamische Beziehungen:

Es reicht selten aus, ausschließlich statische Beziehungen zu verwenden, da bestimmte Signalbelegungen erst Wirkungen zum nächsten Zeitpunkt zur Folge haben. Als Modellform wurde ein getakteter Automat gewählt, der beschreibt, wie die momentanen Werte der Eingangssignale und der momentane Zustand die momentanen Werte der Ausgangssignale und den Folgezustand bewirken. Der aktuelle Zeitpunkt wird mit k markiert, der darauffolgende mit $k+1$:

Werte_der_Eingangssignale(k) & Zustand(k) ==>
Werte_der_Ausgangssignale(k) & Zustand($k+1$).

3. Erweiterung auf Fehlermodelle:

Eine Erweiterung, die einen geringfügig höheren Modellierungsaufwand verursacht, ist die Verwendung von Fehlermodellen. In die Modellformeln werden auf der Bedingungsseite die Fehler mit aufgenommen, d.h., es werden die Fehlerverhalten bei verschiedenen Fehlern mit in das Modell aufgenommen:

Werte_der_Eingangssignale(k) & Zustand(k) & Fehler(k) ==>
Werte_der_Ausgangssignale(k) & Zustand($k+1$).

4. Erweiterung auf unsichere Modelle:

Die kausalen Beziehungen im logischen Modell können um kausal bedingte Wahrscheinlichkeiten ergänzt werden.

Werte_der_Eingangssignale(k) & Zustand(k) & Fehler(k) ==>
Werte_der_Ausgangssignale(k) & Zustand($k+1$)
/ $P(\text{Werte_der_Ausgangssignale}(k) \text{ \& Zustand}(k+1) \mid$
 $\text{Werte_der_Eingangssignale}(k) \text{ \& Zustand}(k) \text{ \& Fehler}(k))$.

Dadurch kann ausgedrückt werden, daß bestimmte Ausgaben nur mit einer gewissen Häufigkeit auftreten.

5. Verwendung von Komponentenbibliotheken und -hierarchien:

Ein Anlagenmodell ist in der Regel komponentenorientiert aufgebaut. Eine Komponente kann dabei repräsentieren

- eine Steuerungs-/Regelungsfunktion, die auf einem Automatisierungsgerät ablaufen soll, oder
- eine physikalische, technologische Prozeßkomponente.

Diese Komponenten sind typisiert, eine konkrete Anlage wird durch Instanzen dieser Typen beschrieben (Typ-Instanz-Konzept).

Weiterhin sind Komponenten hierarchisch strukturiert, d.h. sie können selbst wieder aus Komponenten aufgebaut sein.

Jede Komponente wird durch einen in Form logischer Formeln notierten Automaten beschrieben.

Die Modellierung für die Diagnose ist häufig mit relativ einfachen Mitteln zu bewerkstelligen. Sie kann z.B. parallel zum Steuerungsentwurf erfolgen.

6. Teilung des Diagnosealgorithmus in Off- und On-line-Phase:

Die Aufgabe der Diagnose besteht darin, aus den Meßwerten die im Prozeß aufgetretenen Fehler zu ermitteln:

Diagnosealgorithmus:

Gegeben: Meßwertfolgen,
Modell,

Gesucht: Fehler

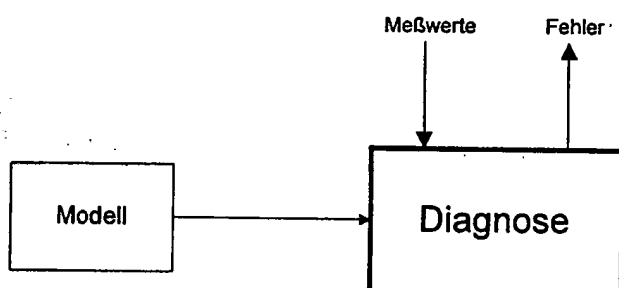


Abbildung 1 Grundschema der Diagnose

Diese Aufgabe kann mit verschiedenen Methoden gelöst werden. Häufig ist die Lösung dieser Aufgabe jedoch zeitkritisch, so daß die zum Diagnosezeitpunkt stattfindenden Berechnungsschritte optimiert werden müssen. Deswegen erfolgt eine Teilung des Diagnosalgorithmus in eine Off-line- und eine On-line-Phase. In der Off-line-Phase wird aus dem Prozeßmodell ein Diagnoseprogramm erzeugt, das unter Echtzeitbedingungen in der On-line-Phase aus den Meßwerten das Schlußfolgern auf Fehler ermöglicht.

In der On-line-Phase wird aus dem Anlagenmodell und entsprechenden Vorgaben für die Hard- und Software-Umgebung der On-line-Phase das Diagnoseprogramm generiert:

Off-line-Phase:

Gegeben: Modell,
Vorgaben für die Hardware- und Software-Umgebung,

Gesucht: Diagnoseprogramm

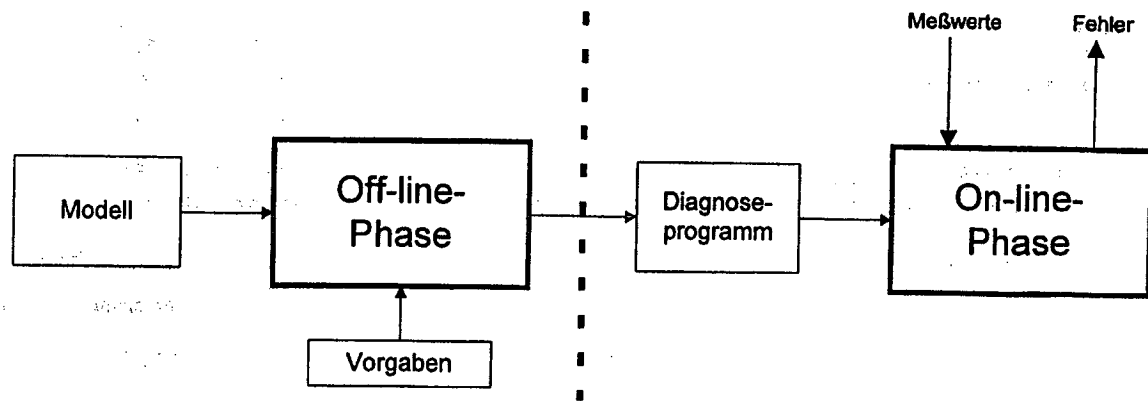


Abbildung 2 Unterteilung der Diagnose in On- und Off-line-Phase

Dieses Diagnoseprogramm führt dann die Diagnose in der Anlagenumgebung aus (On-line-Phase):

On-line-Phase:

Gegeben: Meßwerte,
Diagnoseprogramm,

Gesucht: Fehler

Die Off-line-Phase ist mit der logischen Programmiersprache Prolog realisiert. Das Diagnoseprogramm wird entsprechend der Vorgaben in der Syntax einer imperativen Programmiersprache gebildet. Zur Zeit sind Programme in SCL und C++ erzeugbar.

7. Berechnung von Auftretenswahrscheinlichkeiten von Fehlern:

Anstelle der Ausgabe vieler alternativen Lösungen, die aus Kombinationen von Fehlern bestehen, findet die Berechnung von Auftretenswahrscheinlichkeiten statt.

Die Lösung besteht in der bedingten Wahrscheinlichkeit für jeden Fehler, daß er unter der Bedingung der bisher aufgetretenen Messungen (0..k) auftrat:

$$P(\text{Fehler}(k) \mid \text{Werte_der_Ausgangssignale}(0..k) \ \& \ \text{Werte_der_Eingangssignale}(0..k)) .$$

Als Zwischenergebnis ist dafür die Berechnung von

$$P(\text{Fehler}(k) \ \& \ \text{Zustand}(k+1) \mid \text{Werte_der_Ausgangssignale}(0..k) \ \& \ \text{Werte_der_Eingangssignale}(0..k)) .$$

notwendig, die rekursiv erfolgt.

8. Berechnung der Auftretenswahrscheinlichkeit von Fehlern in der Vergangenheit:

Entsprechend der Prozeßdynamik wirken sich Fehler oft nicht im Moment ihres Auftretens, sondern erst nach einer bestimmten Zeit aus, und sind dann erst aufgrund der Messungen erkennbar. Deswegen wird für jeden Fehler berechnet, mit welcher Wahrscheinlichkeit er irgendwann in der Vergangenheit auftrat:

$P(\text{Fehler}(k) \vee \text{Fehler}(k-1) \vee \dots \vee \text{Fehler}(0) \mid$
 $\text{Werte_der_Ausgangssignale}(0..k) \ \& \ \text{Werte_der_Eingangssignale}(0..k)).$

Als Zwischenergebnis ist dafür die Berechnung von

$P(-\text{Fehler}(k) \ \& \ \dots \ \& \ -\text{Fehler}(0) \ \& \ \text{Zustand}(k+1) \mid$
 $\text{Werte_der_Ausgangssignale}(0..k) \ \& \ \text{Werte_der_Eingangssignale}(0..k) \ \& \ -\text{Fehler}(k-1) \ \& \ \dots \ \& \ -\text{Fehler}(0)).$

notwendig, die rekursiv erfolgt.

9. Berechnung der Auftretenswahrscheinlichkeit des Sammelfehlers:

Als eine Zusammenfassung der Auftretenswahrscheinlichkeiten der Fehler wird die Wahrscheinlichkeit des Sammelfehlers berechnet, d.h. die Wahrscheinlichkeit, daß irgendein Fehler auftrat. Erst wenn diese Wahrscheinlichkeit den Wert 1 annimmt, müssen die Wahrscheinlichkeiten der einzelnen Fehler näher betrachtet werden.

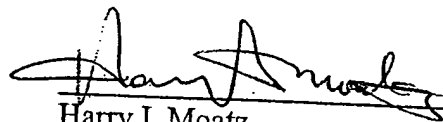
**BEFORE THE OFFICE OF ENROLLMENT AND DISCIPLINE
UNITED STATES PATENT AND TRADEMARK OFFICE**

LIMITED RECOGNITION UNDER 37 CFR § 11.9(b)

Mr. Andreas Horst Lothar Grubert is hereby given limited recognition under 37 CFR §11.9(b) as an employee of Baker Botts LLP, to prepare and prosecute patent applications for clients of Baker Botts LLP in which a member of Baker Botts LLP is the attorney of record. This limited recognition shall expire on the date appearing below, or when whichever of the following events first occurs prior to the date appearing below: (i) Mr. Andreas Horst Lothar Grubert ceases to lawfully reside in the United States, (ii) Mr. Andreas Horst Lothar Grubert's employment with Baker Botts LLP ceases or is terminated, or (iii) Mr. Andreas Horst Lothar Grubert ceases to remain or reside in the United States on an H-1B visa.

This document constitutes proof of such recognition. The original of this document is on file in the Office of Enrollment and Discipline of the U.S. Patent and Trademark Office.

Limited Recognition No. L0225
Expires: June 30, 2006



Harry I. Moatz

Director of Enrollment and Discipline

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.